09/671,875

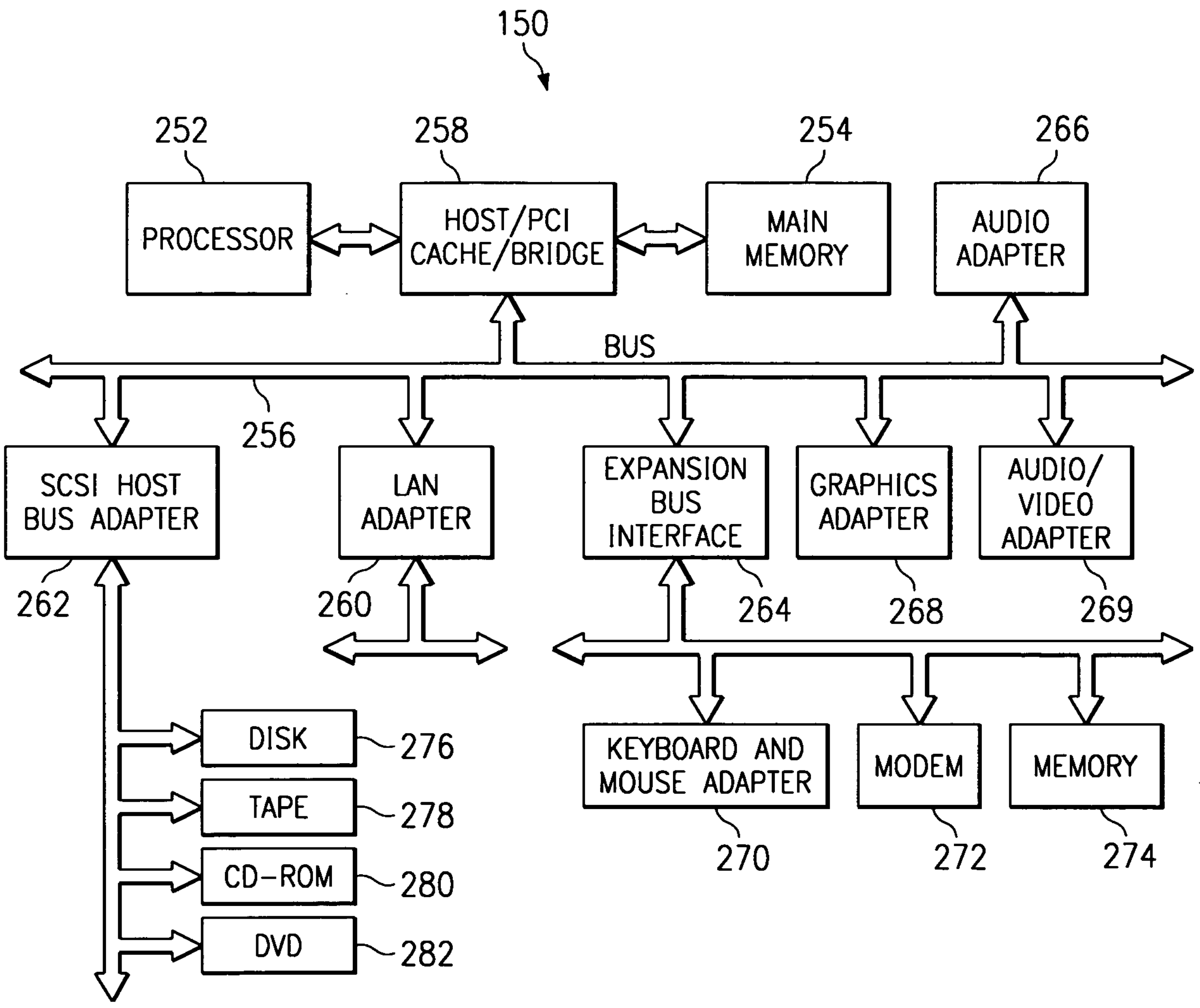

FIG. 1

09/677,875

TEMPLATE	SLOT 0	SLOT 1	SLOT 2
00	M-unit	I-unit	I-unit
01	M-unit	I-unit	I-unit
02	M-unit	I-unit	I-unit
03	M-unit	I-unit	I-unit
04	M-unit	L-unit	X-unit
05	M-unit	L-unit	X-unit
06			
07			
08	M-unit	M-unit	I-unit
09	M-unit	M-unit	I-unit
0A	M-unit	M-unit	I-unit
0B	M-unit	M-unit	I-unit
0C	M-unit	F-unit	I-unit
0D	M-unit	F-unit	I-unit
0E	M-unit	M-unit	F-unit
0F	M-unit	M-unit	F-unit
10	M-unit	I-unit	B-unit
11	M-unit	I-unit	B-unit
12	M-unit	B-unit	B-unit
13	M-unit	B-unit	B-unit
14			
15			
16	B-unit	B-unit	B-unit
17	B-unit	B-unit	B-unit
18	M-unit	M-unit	B-unit
19	M-unit	M-unit	B-unit
1A			
1B			
1C	M-unit	F-unit	B-unit
1D	M-unit	F-unit	B-unit
1E			
1F			

FIG. 2

09/671,875

```
Main Loop:

While there are unprocessed instruction groups {
   Select next instruction group.

TopOfGroup:
   For each instruction in the group:

       switch on instruction_type {
        case TypeA:
              TypesA++;
              TypesMIA++;
           break;
        case TypeM:
              TypesM++;
              TypesMIA++;
           break;
        case TypeI:
              TypesI++;
              TypesMIA++;
           break;
        case TypeB:
              TypesB++;
           break;
        case TypeF:
              TypesF++;
           break;
        case TypeLI:
              TypesLI++;
           break;
        default:
           error;
       }
       TypesALL++;

       /*----------------------------+/
       | Test for the a previous incomplete boundle
       +-----------------------------*/
       if INCOMPLETE is not zero {
        if INCOMPLETE equals M_MI{
          INCOMPLETE = 0;

          if despersal window is large {
MakeM_MI:
              Template = M_MI;
              take-M;
              take-I;
              goto StoreBundle;
          }
          remainder = size of instruction group % 3;

         if remainder = 0 then: {
           if TypesI > 0 AND TypesF < TypesF-units AND TypesM+TypesA <
bundle count then:
              goto MakeM_MI;
           else
              goto MakeMFB;
```

FIG. 3A

09/671,875

```
          }

          if remainder = 1 then: {
             if (TypesI > 0 OR TypesA > 0)  AND TypesF < TypesF-units AND
TypesM+TypesA < bundle count then:
               goto MakeM_MI;
             else
               goto MakeMFB;
          }
          /* remainder = 2 */
          if (TypesI > 0 OR TypesA > 0)  AND TypesF < TypesF-units AND
TypesM+TypesA >= bundle count then:
               goto MakeM_MI;

MakeMFB:
          Template = MFB_;
          nop;
          nopb;
          goto StoreBundle;
        /*-------------------------------------------------------+/
        | INCOMPLETE equals MI_I
        +-------------------------------------------------------*/
        } else {
          INCOMPLETE = 0;
          if despersal window is large {
MakeMI_I:
             Template = MI_I;
             take-I;
             goto StoreBundle;
          }
          remainder = size of instruction group % 3
          if remainder = 2 then:
             goto MakeMIB;

          if remainder = 0 then: {
             if TypesI > 0 AND TypesF < TypesF-units
             AND TypesM+TypesA < bundle count then: goto MakeMI_I;
             else goto MakeMIB;
          }
          /* remainder = 1 */
          if (TypesI > 0 OR TypesA > 0) AND TypesF < TypesF-units
          AND TypesM+TypesA < bundle count then: goto MakeMI_I;

MakeMIB:
             Template = MIB_;
             nopB;
             goto StoreBundle;
        }
     }
 While TypesALL > 0} // while instructions remain in group
```

FIG. 3B

```
        if TypesALL is equal to TypesMIA {
         if TypesALL > 3 {
            if TypesM > TypesI+TypesA then: Template = MMI; take-M;take-M;take-I
            else template = MII; take-M;take-I;take-I
            goto StoreBundle;
         }
         if TypesALL = 3 and this is the first bundle of the group{
            if TypesI > 1 then: Template = MII; takeM;take-I;take-I
            else template = MMI; take-M;take-I;take-I
            goto StoreBundle;
         }
         if TypesALL = 2 OR TypesALL = 1 and TypesI = 1}
            if TypesM = 2 then: Template = MMF; take-M;take-M;nop; goto
StoreBundle;
            else INCOMPLETE = MI_I take-M; take-I; goto TopOfGroup
         }
         /* TypesALL = 1 */
             INCOMPLETE = M_MI take-M; goto TopOfGroup
        }
        if TypesLX > 0 then: Template= MLX; take-M; take-LX; goto StoreBundle;

        if TypesB > 0 AND TypesALL-TypesB < 3 then: {
         if typesF > 0 then: {
            if TypesF+TypesI = 2 then: Template= MFI nop;takeF;take-I; goto
StoreBundle;
            else Template=MFB take-M;takeF;take-B; goto StoreBundle;
         } else if TypesI > 0 {
            if TypesI = 2 then: Template= MII  nop;takeF;take-I; goto StoreBundle;
            else Template=MIB take-M;take-I;take-B; goto StoreBundle;
         }else if TypesM = 2 then: Template = MMB take-M;take-M;take-B; goto
StoreBundle;
        else if TypesALL-TypesB = 2 then: Template = MIB take-M;take-I;take-B;
goto StoreBundle;
        else if TypesB = 1 then: Template = MFB take-M;takeF;take-B; goto
StoreBundle;
        else if there are 2 TypesB instructions  or 1 non-TypesB: Template = MBB
take-M;take-B;take-B; goto StoreBundle;
        else Template = BBB take-B;take-B;take-B; goto StoreBundle;
        }
        /* TypesF > 0 */
         if TypesALL = 3 AND TypesM = 2 then: Template = MMF take-M;take-M takeF;
goto StoreBundle;
         else Template = MFI take-M;takeF take-I; goto StoreBundle;
}
storeBundle:
   if TypesALL = 0 then: insert stop bit in Template;
   build bundle in code buffer;
   if TypesALL = 0 then: goto MainLoop;
   else goto TopOGroup;
}
DONE
   if INCOMPLETE is not zero then:      {
        if INCOMPLETE = M_MI then: Template = MFB; nop; nopB; goto StoreBundle;
        else Template = MIB; nopB; goto StoreBundle;
   }
```

FIG. 3C

09/671,875

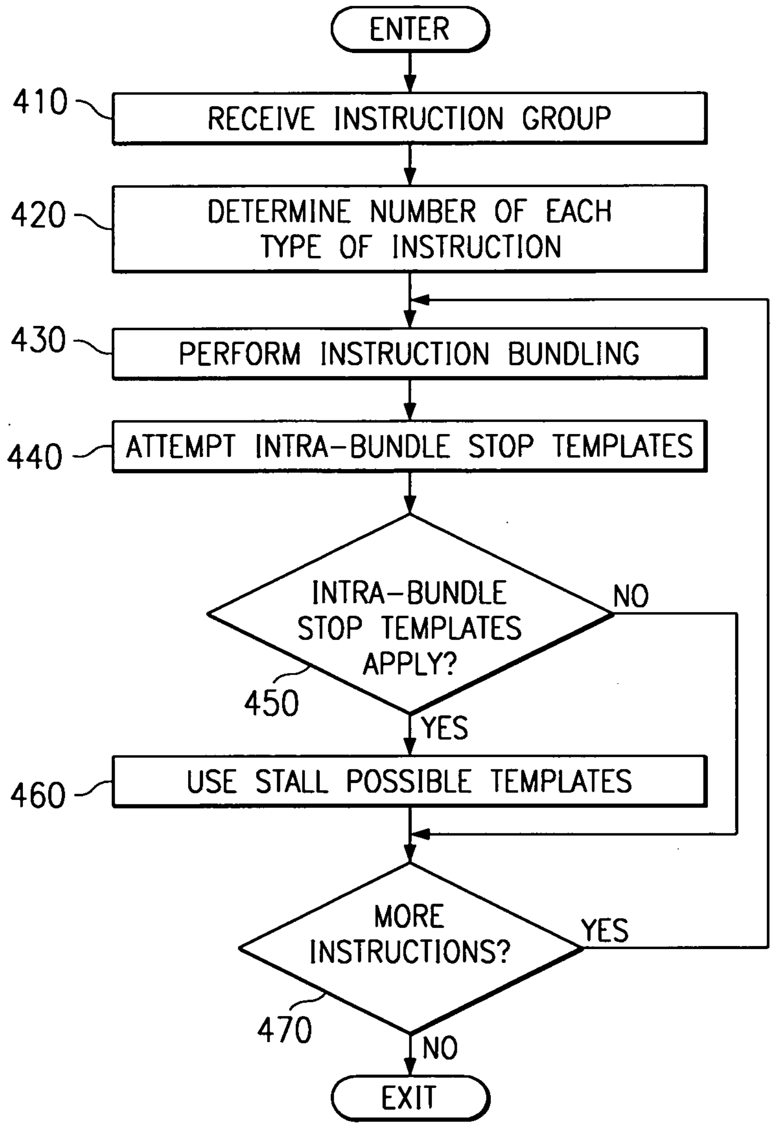
ENTER
410
RECEIVE INSTRUCTION GROUP
420
DETERMINE NUMBER OF EACH TYPE OF INSTRUCTION
430
PERFORM INSTRUCTION BUNDLING
440
ATTEMPT INTRA-BUNDLE STOP TEMPLATES
INTRA-BUNDLE STOP TEMPLATES APPLY?
450
NO
YES
460
USE STALL POSSIBLE TEMPLATES
MORE INSTRUCTIONS?
470
YES
NO
EXIT

FIG. 4